
Appendix D: UNIX diagnostics

This appendix lists the available UNIX-based diagnostic utilities developed by Nortel for IVR 2.0/I.

accdiag

The accdiag utility analyses the status of ACCESS-related components in the IVR system. The **accdiag** command is in the **/u/ivr/vrs/exe** directory and must be run from that directory.

To test the ACCESS link, type **accdiag** in a UNIX shell window.

The following option flags are available with the **accdiag** command:

```
[ -l LogFileName]
[ -n LogicalLinkNumber]
[ ConfigFilePath]
[ ?]
```

The *System Status* screen appears. The following options are available through the System Status screen:

Table D-1
System Status window function keys

Function key	Explanation
F1 Start lh	This option starts the link handler.

Table D-1
System Status window function keys (Continued)

Function key	Explanation
F2 Om Data	This option calls the System Operational Measurements window, as shown Table D-2 .
F3	This option redraws the screen.
F4	This option exits the system.

The System Operational Measurements window includes the following data: Packet Counts, Error Counts, and Link Handler statistics.

[Table D-2](#) outlines function key options available in the System Operational Measurements window.

Table D-2
System Operational Measurements function key options

Function key	Explanation
F2 Clear OM	Clears existing Measurements from window.
F3 Redraw	Refreshes the System Operational Measurements window.
F4	Exits the window.

backup_default.bup file

The backup_default.bup file contains the default list of files used by the backup utility. This list can be edited.

To view the backup_default.bup file, type

```
more /u/ivrr/backup/backup_default.bup
```

at the UNIX shell prompt.

The default contents of this file are the */u/ivr/apps* and */u/ivr/log.d* directories. The other directories and files in the *backup_default.bup* file are dependent on the features installed in your system. Other directories you may want to backup include the following:

Local database-related files

/u/ivr/sys_files/.dbm*
/u/ivr/sys_files/.tpl*
/u/ivr/sys_files/.ext*

Host connectivity (except vt100)

*/u/ivr/3270/**
/u/ivr/sys_files/vtekhos_tadm

vt100

*/u/ivr/vt100/**

System configuration

/u/ivr/exe/.sai*
/u/ivr/sys_files/sysgen.d
/u/ivr/vrs/exe/lh.config

User functions

*/u/ivr/usr/**

Backup list file

/u/ivr/backup/backup_default.bup

Notes:

- 1** Host connectivity files created by the *express_adm* tool should also be backed up using *express_adm*. For information on backing up these files, refer to the *Apertus EXPRESS Administration Guide*. (The *express_adm* tool is used for configuring the Arnet Sync/570 adapter and Madge Token Ring adapter for 3270/5250 host connectivity.)
- 2** The backup tool should not be used to backup system files.

dappl

The `dappl` command is used to dump the content and call flow of applications.

This command generates two report listings:

- The “Listing of the Application” file provides specific configuration details for each cell in ASCII format.
- The “Graphical Structure Chart” file provides a representation of the call flow in ASCII format.

The `dappl` command is found in the `/u/ivr/tools` directory and must be run from this directory or from a directory at the same level. To invoke the `dappl` command, type the following:

```
dappl appl_name
```

where *appl_name* is the name of the application to be dumped.

Table D-3 contains the flags available for the `dappl` command.

Table D-3
Flags for `dappl` command

Flag	Explanation
-h	Display help message
-c	Cell contents output (default setting)
-nc	No cell contents output
-g	Graphic output (default setting)
-s	Only success branches in graphic output
-ng	No Graphic Output
-d	Debugging (full) dump
-nd	No Debugging (default setting)
-cwd	Read application from . rather than ../apps
<appl>	Name of application to dump (REQUIRED) (Do not include path, extension is optional)

Directing the dappl report to a file

You can direct the results of the dappl command to a file. To direct the result of running the dappl command for the application called “test” to a file, enter the following command:

```
dappl test > test.out
```

Viewing the dappl output file

To view the contents of the file *test.out*, enter the following command:

```
more test.out
```

The following figures show the results of the dappl command.

Figure D-1
dappl command application listing output

```
Listing of Application ../apps/test.vpf
Application Revision : 3
Editor Release      : 1.3

Application contains 8 cells:
  Uses NO Databases
  Uses NO User Buffers

Cell 0: DFLT
  Database           : <NONE>
  Pause Digit        : 6
  Resume Digit       : 7
  Termination Digit  : #
  Abort Digit        : NONE
  Pause Time         : 15
  Pause Action       : ABORT
  Interdigit Timeout : 5
  Pause Prompt       : 65
  Resume Prompt      : 115
  Abort Prompt       : 117
  Min Voice Duration : 200
  Cleanup            : <NONE>
  Next Cell          : 2 (Untitled #2 (ANSW))
  SQL DBMS Type      : <NONE>
  SQL Database       : <NONE>
  SQL Server Count   : 1

Cell 1: HANG|
  Cell Description   : Hangup
  Name              : hangup #1 (HANG)
  Type              : HANG
  Prompt Count      : 1
  Prompt            : 55, 0, 0, 0, 0
```



```

Cell 7: HANG
  Cell Description      : Hangup
  Name                  : Error hangup #7 (HANG)
  Type                  : HANG
  Prompt Count          : 1
  Prompt                : 73, 0, 0, 0, 0

```

Figure D-2
dapfl command graphical structure chart output

```

Graphical Structure Chart for Application 'test'

Cell_0:DFLT:'c
../cells/delv.cdsc
../Cell'
+-->Next Cell:
  Cell_2:ANSW:'Answer #2 (ANSW)'
  +-->Next Cell:
    Cell_3:PDAT:'Play IVR # #3 (PDAT)'
    +-->Next Error:
      | Cell_7:HANG:'Error hangup #7 (HANG)'
    +-->Next Cell:
      Cell_4:PDAT:'Play orig. # #4 (PDAT)'
      +-->Next Error:
        | Cell_7:HANG:'Error hangup #7 (HANG)'
      +-->Next Cell:
        Cell_5:GDAT:'Get DTMF #5 (GDAT)'
        +-->Next Error:
          | Cell_7:HANG:'Error hangup #7 (HANG)'
        +-->Next Timeout:
          | Cell_7:HANG:'Error hangup #7 (HANG)'
        +-->Next Cell:
          Cell_6:PDAT:'Play # #6 (PDAT)'
          +-->Next Error:
            | Cell_7:HANG:'Error hangup #7 (HANG)'
          +-->Next Cell:
            Cell_1:HANG:'hangup #1 (HANG)'

Integrity Check:
All cells in application have been referenced.
(EOF):

```

System Applications Monitor

The System Applications Monitor (SAM) utility is a stand-alone process that logs user activity as an application runs on one or more channels. This online application debugging tool tracks the execution path of the application and captures all selected data.

Table D-4 defines the data types accessed and captured by the **sam** tool.

Table D-4
sam data capture list

Data type	Definition
Channel number	Identifies the channel on which the cell has been executed.
Application name	Name of the application being processed.
Application ID	Unique number identifying the application.
Current cell type	Type of cell that was executed (PLAY, MENU).
Current cell number	Unique number identifying the cell in the application.
Next cell type	The next cell type.
Next cell number	Unique number identifying the next cell in the application.
Digits collected	String of digits input by the caller for the current cell.

Typically you use **sam** to monitor a call, tracing the transition between cells and identifying caller-selected data.

Note: Although you can monitor multiple channels, it is recommended that you run **sam** on only one channel at a time to avoid compromising system performance.

If two digits are collected during cell execution, **sam** generates two log entries for the same cell.

The first entry identifies the digits. You can disregard the Next Cell Type and Next Cell Number fields, which **sam** logs correspondingly as blank and “-1”. The second entry identifies the next cell (see [Table D-4](#)).

Accessing sam

The **sam** utility is found in the `/u/ivr/exe` directory, but can be accessed from any directory.

To invoke **sam**, enter the following at the shell prompt:

```
sam -f filename -lo low_channel_num
    -hi high_channel_num
```

[Table D-5](#) defines the variables used in the **sam** utility.

Table D-5
sam command line variables

Variable	Definition
<i>filename</i>	Identifies the name of the file where you want sam to log the information. The default filename is <code>sam.out</code> in the current directory.
<i>low_channel_num</i>	Shows the low end of the channel range that you want to monitor. If you do not specify a number, the default value is 0.
<i>high_channel_num</i>	Shows the high end of the channel range that you want to monitor. The number for this parameter must be equal to or less than 48. If you do not specify a number, the default value is 63. Do not use this default value. In Meridian IVR, a maximum number of 48 channels are available.

Example scenario

To log the application **mikefl** running on channel 2, enter the following command:

```
sam -f -lo 2 -hi 2
```

Since a file name was not entered, the **sam** utility will log user activity to the default file *sam.out*.

Table D-6
sam sample output

Channel: 2 Appl Name: mikefl Appl_id: 0 Cur_cell_type: DFLT Cur_cell_num: 0 Next_cell_type: ANSW Next_cell_num: 1 Digits:	← indicates application start t
Channel: 2 Appl Name: mikefl Appl_id: 0 Cur_cell_type: ANSW Cur_cell_num: 1 Next_cell_type: MENU Next_cell_num: 2 Digits:	
Channel: 2 Appl Name: mikefl Appl_id: 0 Cur_cell_type: MENU Cur_cell_num: 2 Next_cell_type: Next_cell_num: -1 Digits: 2	Notice these two entries in the log correspond to the same current cell. The first entry identifies the digits that are captured; the next cell is indicated as "-1".
Channel: 2 Appl Name: mikefl Appl_id: 0 Cur_cell_type: MENU Cur_cell_num: 2 Next_cell_type: PLAY Next_cell_num: 4 Digits:	The second entry identifies the Next cell type and number.

Exiting sam

The **sam** utility runs on the specified range of channels until you enter <Ctrl + c> to end the process. At that point, **sam** stops writing channel activity data and returns control to the shell.

Displaying and printing the SAM log

The log generated by **sam** is an ASCII file with the default name of *sam.out*.

To display the log, enter this shell command:

```
pg /u/ivr/exe/sam.out
```

To print the log, enter this shell command:

```
lp /u/ivr/exe/sam.out
```

It can be useful to compare the **sam** log file to your application callflow.

The flags in [Table D-7](#) can be used with the **sam** command.

Table D-7
sam command flags

Flag	Explanation
[-verbose]	for verbose mode
[-f <i>filename</i>]	to write application data to a file
[-lo <i>channel##</i>]	for low range of trace channels
[-hi <i>channel##</i>]	for high range of trace channels
[-b <i>branch</i>]	specifies a MIVR/I to attach to
[-r]	to report revision level of sam utility

Sample output

The following is an example of output from the **sam** command.

```
*** SAM Trace Started on channels 0 to 0 ***
*** SAM Trace Stopped ***
```

devnt

The **devnt** utility displays a list of all scheduled application times, telephone numbers, and event IDs, as contained in the */sys_files/csc_events* directory. The **devnt** command must be run from the */u/ivr/sys_files* directory.

dsysf

The **dsysf** tool displays the contents of the system configuration file.

Sample output

The following is an example of output from the **dsysf** command.

```
SYSTEM DATA
system_data_size      60
channel_data_size     80
node_data_size        100
sysgen_version        2
sysgen_update         0
total_nodes           1
max_system_prompt     0x7D0
hisys_vsn             0x7CF
backup_vsn            0xF601
duplicate log path    /dev/null
NODE 1 DATA
node_type              SYS_TYPE_MERIDIAN
max_channels          12
active_channels       12
connection_type       SYS_CONNECT_ETHERNET
node_name             localhost
CHANNEL 0 DATA
channel_type           SYS_TTYPE_SHARED
channel_init_type     SYS_ITYPE_ANI2
channel_dir           SYS_DIRECT_IN
channel_mbox          2000
channel_pwd           200000
channel_class         0
access_link           1
CHANNEL 1 DATA
channel_type           SYS_TTYPE_SHARED
channel_init_type     SYS_ITYPE_ANI2
channel_dir           SYS_DIRECT_IN
channel_mbox          2000
channel_pwd           200000
channel_class         0
access_link           1
```

CHANNEL 2 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1
CHANNEL 3 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1
CHANNEL 4 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1
CHANNEL 5 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1
CHANNEL 6 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1
CHANNEL 7 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1

CHANNEL 8 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1
CHANNEL 9 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1
CHANNEL 10 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1
CHANNEL 11 DATA	
channel_type	SYS_TTYPE_SHARED
channel_init_type	SYS_ITYPE_ANI2
channel_dir	SYS_DIRECT_IN
channel_mbox	2000
channel_pwd	200000
channel_class	0
access_link	1

kccheck

The **kccheck** tool displays the list of Meridian IVR options available on your system.

The following is an example of output from the **kccheck** command.

```
Meridian IVR Option List
=====
Version number           : 2.00
Platform                 : SCO UNIX
Voice server             : Other than MM Opt 11
Development options      : Full
Fax response             : yes

3270 SNA host connectivity : yes
3270 QLLC host connectivity : yes
5250 emulation           : yes
X.25 host connectivity    : yes
VT100 host connectivity   : yes
Informix connectivity     : yes
Oracle connectivity       : yes
Ingres connectivity       : yes
Sybase connectivity       : yes
Smart UPS                 : yes

Number of IVR ports       : 96
Number of fax ports       : 8
=====
```

readdongle

The **readdongle** command checks the status of the security keylock. If the keylock is properly installed, its serial number will be displayed as output.

If the keylock is not properly installed or is missing, the following message appears:

```
Failed to read from the security keylock device
```

Transaction logs

Transaction logs are files into which errors and transactions may be written. These files reside in the */u/ivr/log.d* directory.

The following files contain transaction data:

- event.log
- sec.log
- start.log
- xconsole.log

Event log

The **event.log** file contains a timestamped record of all application events.

To access the event log, complete the following procedure:

Procedure D-1 Accessing the event log

- 1 At a UNIX shell prompt, change to the */u/ivr/log.d* directory using

```
cd /u/ivr/log.d
```
- 2 Type the following:

```
cat event.log
```

The event log is displayed as shown in the section below.

Sample output

```
09/21 11:08 CLI ADVISORY 6 Application Load : eight
09/21 11:08 CLI ADVISORY 10 Application Startup : eight
09/21 11:10 CLI ADVISORY 71 Stopping Application: eight
09/21 11:10 CLI ADVISORY 72 Application Stopped: eight
09/21 11:10 CLI ADVISORY 12 Application Unload : eight
09/21 11:10 CLI ADVISORY 6 Application Load : test1
09/21 11:10 CLI ADVISORY 10 Application Startup : test1
```


Sec log

The **sec.log** file contains all security-related messages from the application.

To access the sec log, complete the following procedure.

Procedure D-2

Accessing the sec log

- 1 Change to the `/u/ivr/log.d` directory at a UNIX shell prompt using
cd /u/ivr/log.d
- 2 Type the following
more sec.log

The sec log is displayed as shown in the section below.

Sample output

```
**** Error Messages from process: vrted ****  
  
**** Error Messages from process: vrted ****  
  
**** Error Messages from process: vrted ****
```

Start log

The **start.log** file contains a timestamped record of all application and tool start times.

To access the start log, complete the procedure below.

Procedure D-3

Accessing the start log

- 1 At a UNIX shell, change to the `/u/ivr/log.d` directory using
cd /u/ivr/log.d
- 2 Type the following:
cat start.log

The start.log file is displayed, as shown in the section below.

Sample output

```
[09/21 11:00] System Started
[09/21 11:00] ./ueh p[5130] q[224] Running
[09/21 11:00] ./vtk p[5131] q[232] Running
[09/21 11:00] ./vip p[5132] q[227] Running
[09/21 11:00] ./vrm p[5133] q[230] Running
[09/21 11:00] ./cli p[5134] q[231] Running
[09/21 11:00] ./dbs p[5135] q[228] Running
[09/21 11:00] ./qds p[5136] q[222] Running
[09/21 11:00] ./csc p[5137] q[223] Running
[09/21 11:00] ./sad p[5138] q[229] Running
[09/21 11:00] ./vft p[5139] q[225] Running
[09/21 11:00] ./pmg p[5140] q[226] Running
[09/21 11:00] ./trs p[5141] q[233] Running
[09/21 11:00] ./ust p[5142] q[221] Running
```

Xconsole log

The xconsole.log file contains a timestamped record of all events logged through the console.

To access the xconsole log, complete the procedure below

Procedure D-4

Accessing the xconsole log

- 1 At a UNIX shell prompt, change to the `/u/ivr/log.d` directory using
cd /u/ivr/log.d
- 2 Type the following:
more xconsole.log

The xconsole.log file appears, as shown in the example below.

Sample output

```
07/28 15:00 Welcome to Meridian IVR/I. Console Ready.
08/03 18:37 Starting Meridian IVR/I...
08/03 18:38 Meridian IVR/I startup complete, assuming the link is up
08/03 18:38 Error reading the security profile
08/04 09:57 Starting Meridian IVR/I...
08/04 09:58 Meridian IVR/I startup complete, assuming the link is up
08/04 10:01 Unable to open software profile.
07/28 15:12 CLI ADVISORY 10 Application Startup : test1
07/28 16:52 CLI ADVISORY 71 Stopping Application: test1
07/28 16:52 CLI MINOR 72 Application Stopped: test1
07/28 16:52 CLI ADVISORY 12 Application Unload : test1
07/28 16:54 Stopping Meridian IVR/I...
```

```
07/28 16:54 Meridian IVR/I is down
07/28 16:54 Starting Meridian IVR/I...
07/28 16:55 Meridian IVR/I startup complete, assuming the link is up
07/28 16:55 VIP1      ADVISORY 100  Process Startup
07/28 16:55 Generations process ./trs terminated prematurely.
07/28 16:55 CLI      ADVISORY 0    CLI Startup
07/28 16:55 DBS      ADVISORY 800  Process Startup
07/28 16:55 CSC      ADVISORY 1100 Process Startup
07/28 16:55 SAD      ADVISORY 1300 Startup
```

rstat

The **rstat** command resets the call audit, cell detail, channel usage, and subscriber statistics to zero.

To run the **rstat** command, type

rstat

at a UNIX shell prompt.

*The following confirmation message appears after you run the **rstat** command:*

```
Statistics Have Been Reset.
```

This command is also available through the IVR menu interface.

sched

The **sched** utility manipulates and schedules events. This command must be run from the **/u/ivr/sys_files** directory.

Command flags

The following flags are available with the sched command.

Usage:

```
sched -d
-a application_name
[-v]
[-o outdial_number]
{-t abs_time | -r n | -n n,mmddyyyymm}
[-h handle] [-l aa ... -5 aa]
```

OR

```
2] sched -u
-e event_id [-v]
```

OR

```
3] sched -l
-a application_name -o outdial_number [-v]
```

OR

```
4] sched -r {f,s}
```

testivr.vpf

The testivr.vpf application in the **/u/ivr/apps** directory is used as a sanity check after installation. It is run from Application Management.

testfax.vpf

The testfax.vpf application in the **/u/ivr/apps** directory is used as a sanity check for fax after installation. It is run from Application Management.

Vrs tools

The following vrs tools are accessed through the */u/ivr/vrs/tools* directory:

- dpsm
- gmessage
- maudit

dpsm

The **dpsm** utility displays a channel map for the system.

Command flags

The following flags are available with the dpsm command.

Usage:

```
dpsm [-branch name] to specify the branch to use  
[-audit chan] to specify a channel to audit  
[-chan] to display a channel map  
[-log] to display the alarm log
```

gmessage

The **gmessage** utility is used by the system and is not intended for use by the user.

maudit

The **maudit** utility displays the audit trail for a particular channel.

Command flags

The following flags are available with the maudit command.

Usage:

```
maudit [-channel #] for channel number  
[-branch path] to specify the IVR branch
```